Converting Transaction Records into Longitudinal Data Sets

Carl Formoso
Division of Child Support
Washington State
May 2002

Abstract

Two problems are used to illustrate methods we have used to create longitudinal data for analytical purposes.

In the first, and simpler, problem all the transaction data are in one file and we use an array with a calculated index to properly position each record in the longitudinal file. The transaction file must be summarized to one transaction per individual in the time unit of interest, and must be sorted on the individual client identification variable. However, because the index is being calculated the transactions can be presented in any time order, and it is easy to select a time range of interest for the longitudinal file.

In the second problem, the transaction records for a given time, the particular month in our case, are found in several historical transaction files, with a variable number of transactions files required to obtain complete records. In this problem we make use of nested macro variables to create an automated procedure which will select the appropriate transaction record files and collect records into the proper time slots. The procedure first creates very small data sets simply containing a list of contents, in terms of month of transaction, of the very large transaction files. Then, by reading the small contents files, a macro variable is created containing a list of the data set names which are required to obtain complete records for a given month. Only the required large transaction files are read to create a file with only records for the particular month, and duplicate records are eliminated. The monthly files are summed by individual and then concatenated to create a longitudinal data set.

Using a Calculated Index

In this work we began with wage data from Washington State Employment Security Department, which has one record of quarterly earnings for each employee-employer combination. Since we are only

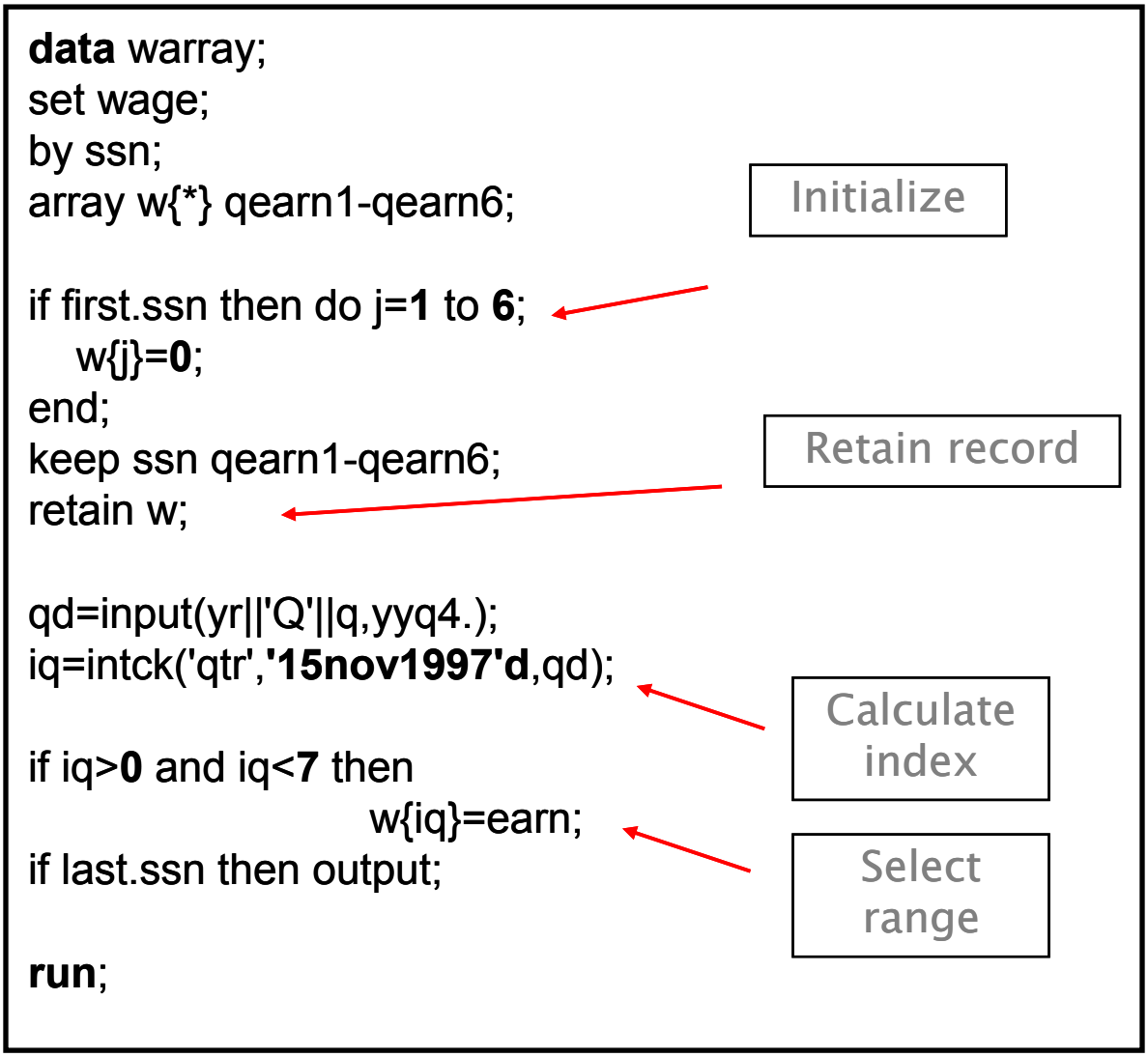
interested in employee earnings, quarterly earnings are summed for each wage earner as shown in Table 1. This is actual data for a sample of five individuals with the identifying SSNs coded.

Table 1: Quarterly Earnings Records for Five Individuals

SSN	yr (char)	q (char)	earn (¢)
A	98	2	32292
A	99	1	132252
A	98	4	22757
A	98	3	162190
B	98	2	40700
B	98	3	136760
B	98	4	3700
C	97	1	424589
C	98	1	549161
C	98	2	621401
C	98	3	843038
D	98	1	615685
D	98	2	595613
D	98	3	643091
E	98	1	352106
E	99	1	277727
E	99	2	96116

In this example we want to create a longitudinal file with earnings records from 98Q1 to 99Q2. Note that in Table 1 the first record for client "C" is outside this range and that records for client "A" are out of sequence. The code to convert to a longitudinal record is shown in Figure 1. The coding requires that the input file be sorted on SSN and that there be no more than one record per client for each quarter.

Figure 1: Coding for Calculated Index Conversion



On the first observation with a new SSN the wage array is initialized to zero to eliminate the previous record and so that quarters with no earning are recorded as 0 rather than missing. Then the longitudinal record of earnings must be retained while the quarterly observations are read in for each SSN. The array index is calculated by converting the character values for year and quarter to a SAS date value and calculating the number of quarters difference from a reference date. If the index falls into the desired range the earnings observation is read into the wage array. On the final observation for a given SSN the longitudinal record is written to the new data set.

The coding produces the longitudinal data set shown in Table 2. The variable names could be replaced with more meaningful names by the techniques described in the next section of this paper.

Table 2: Longitudinal Earnings Data

SSN	qearn1	qearn2	qearn3	qearn4	qearn5	qearn6
A	0	32292	162190	22757	132252	0
B	0	40700	136760	3700	0	0
C	549161	621401	843038	0	0	0
D	615685	595613	643091	0	0	0
E	352106	0	0	0	277727	96116

Using Nested Macro Variables

In this work we needed to convert archived files for child support payments into a longitudinal data file covering 60 months from January 1997 to December 2001. However, the way the data was archived was not consistent across this 5 year time period. Payments made in a particular month might be found in 1, or up to 5, archived files. This is demonstrated for the beginning of 1998 in Table 3. While many of the payments for a given month stored in multiple archived files are duplicates, not all are. So all archived files containing payment records for a given month must be examined.

We approached this problem by first creating a set of macro variables for file reference, shown in Figure 2, and a set of macro variables for payment reference, shown in Figure 3.

The macro, ***paydistc***, shown in Figure 4, reads all of the large archived files from library ***payf*** and creates very small files listing the payment months contained in each large file. The resolution of a nested macro variable is shown in Figure 5 for the jan98 archived file, and the resulting coding of ***paydistc*** for jan98 is shown in Figure 6. The contents list file is shown in Figure 7 where we see that the jan98 archived file contains payment records for January 1998 and February 1998 (see Table 3 for comparison).

Table 3: Sample Child Support Payments Archived Data

Payment Records in Archived Data File for Month

Payment Made in Month

	1998								
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	S
Jan	243,644	243,645	243,645						
Feb	42,816	229,611	229,610						
Mar		44,027	267,161						
Apr			44,092	264,377	264,377	264,377			
May				28,116	248,274	248,274			
Jun					56,748	264,422			
Jul						33,812	258,866	258,867	
Aug							34,134	233,300	
Sep								58,499	
Oct									

Figure 2: Macro Variables for File Reference

```
%let m1=jan;
%let m2=feb;
%let m3=mar;
%let m4=apr;
%let m5=may;
%let m6=jun;
%let m7=jul;
%let m8=aug;
%let m9=sep;
%let m10=oct;
%let m11=nov;
%let m12=dec;

%let y1=97;
%let y2=98;
%let y3=99;
%let y4=00;
%let y5=01;
```

Figure 3: Macro Variables for Payment Reference

```
%let pm1=01;
%let pm2=02;
%let pm3=03;
%let pm4=04;
%let pm5=05;
%let pm6=06;
%let pm7=07;
%let pm8=08;
%let pm9=09;
%let pm10=10;
%let pm11=11;
%let pm12=12;

%let py1=1997;
%let py2=1998;
%let py3=1999;
%let py4=2000;
%let py5=2001;
```

Figure 3: Macro to Create Contents Lists

```
%macro payfdistc;

%do yr=1 %to 5;
   %do mo=1 %to 12;
        create table   &&m&mo.&&y&yr    as select distinct
          substr(dte,1,4) as pyr, substr(dte,5,2) as pmo,
                yr as fyr, mo as fmo
        from      payf.&&m&mo.&&y&yr;


   %end;
%end;

%mend;
proc sql;
%payfdistc
```

Figure 5: Macro Variable Resolution

With yr=2 and mo=1

&&m&mo.&&y&yr

first pass

&m1.&y2

second pass

jan98

Figure 6: Create Contents List for Jan98 Archived File

```
proc sql;

create table jan98 as select distinct
            substr(dte,1,4) as pyr, substr(dte,5,2) as pmo,
                  yr as fyr, mo as fmo
      from      payf.jan98;
```

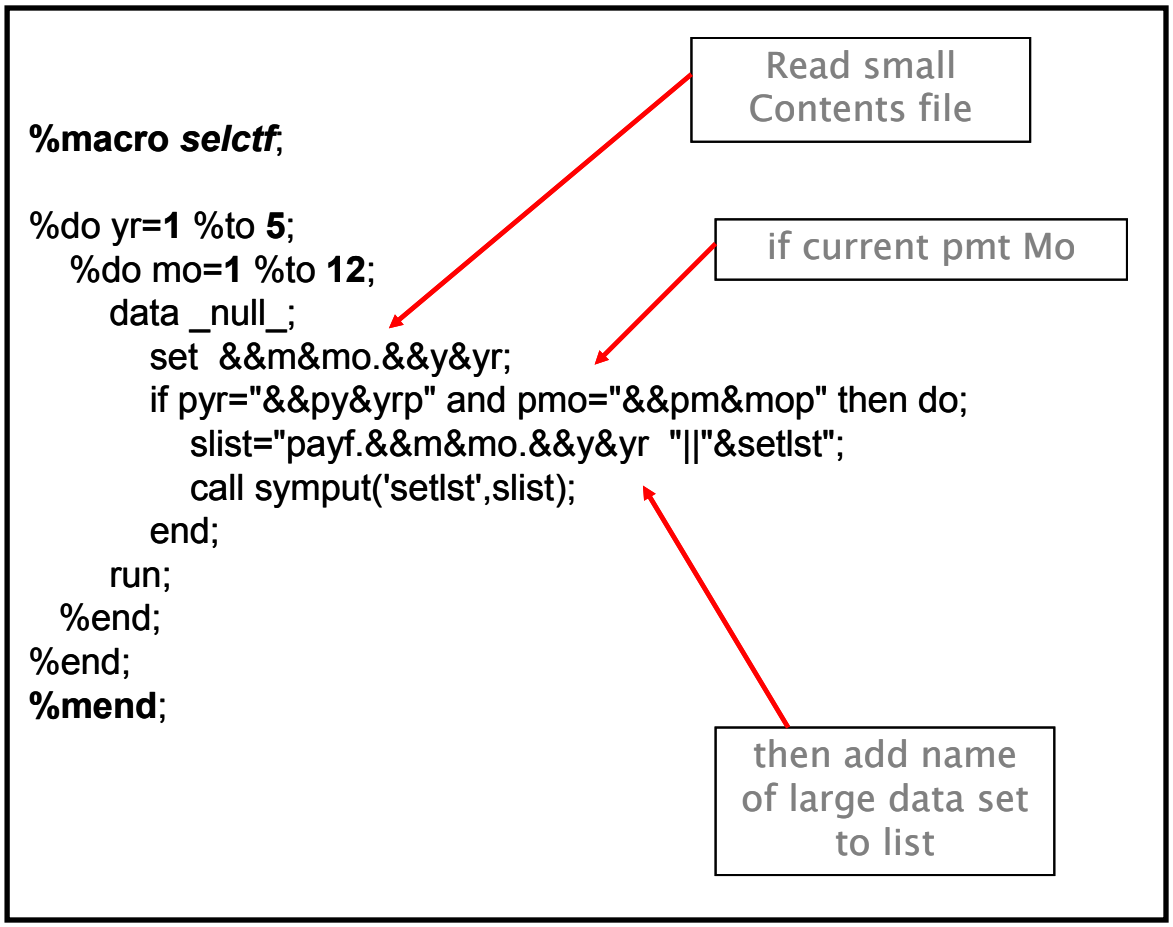
Figure 7: Contents List for Jan98 Archived File

pyr	pmo	fyr	fmo
1998	01	98	01
1998	02	98	01

The macro ***selctf***, shown in Figure 8, then reads all the very small contents files to obtain the names of the archived files which contain payment records for a particular month. The macro creates a macro variable, ***setlist***, which lists all the names of the large archived files which must be read to obtain complete payment records for a given

month. ***Selctf*** is called from the macro ***monthf***, shown in Figure 9. For each of the 60 months in the longitudinal record the macro variable ***setlst*** is first set to null to eliminate the previous list. Then ***selctf*** creates a list appropriate for the particular payment month. The required large archived files are read and the payment records for the particular month are extracted. Duplicates are eliminated and a data set is created containing complete payment records for the particular month.

Figure 8: Macro to Create List of Archived File Names




```
%macro selctf;

%do yr=1 %to 5;
   %do mo=1 %to 12;
      data _null_;
         set  &&m&mo.&&y&yr;
         if pyr="&&py&yrp" and pmo="&&pm&mop" then do;
            slist="payf.&&m&mo.&&y&yr  "||"&setlst";
            call symput('setlst',slist);
         end;
      run;
   %end;
%end;
%mend;
```

Figure 9: Macro to Create Payment Data Sets for Each Month

```
%macro monthf;
%do yrp=1 %to 5;
   %do mop=1 %to 12;
      %let setlst=  ;
                    %selctf

      data p&&py&yrp.&&pm&mop;
         set &setlst;
         if substr(dte,1,4)="&&py&yrp" and substr(dte,5,2)="&&pm&mop";
         drop yr mo;
      run;

      proc sql;
         create table payf.p&&py&yrp.&&pm&mop as
                  select distinct *  from p&&py&yrp.&&pm&mop;
      quit;
   %end;
%end;
%mend;

%monthf
```

Reads appropriate data sets, and extracts payments for a single month

Eliminates dups

This procedure creates 60 data sets, each one containing complete payment records for a single month. After monthly summation by individual the 60 data sets are concatenated to create the longitudinal data set with payment histories across 60 months for each individual. A sample with coded identification is shown in Figure 10.

Figure 10: Longitudinal Payment (¢) History Data

	Jan97	Feb97	Mar97	Apr97	May97	Jun97	
Al	7500	3750	3750	7500	7500	8750	
Bob	0	0	0	10000	0	0	
Chuck	0	0	0	2966	0	3580	… etc.
Doris	0	10000	0	5000	12500	5000	
Ed	14305	14295	12865	12886	12875	0	

etc.